

Budapesti Műszaki és Gazdaságtudományi Egyetem  
Villamosmérnöki és Informatikai Kar  
Irányítástechnika és Informatika Tanszék

# Open source Android alkalmazás közösségi fejlesztése

## TÉMALABORATÓRIUM BESZÁMOLÓ

Főglein Simon István  
ZA0D8T

Konzulens: Dr. Somogyi Péter

2022. november 23.

# Tartalomjegyzék

<b>Tartalomjegyzék</b>	<b>i</b>
<b>Kivonat</b>	<b>a</b>
<b>1. Bevezetés, célkitűzések</b>	<b>1</b>
<b>2. Androidos SplashScreen megoldások ismertetése</b>	<b>2</b>
2.1. API level 30 (Android 11) és korábbi . . . . .	2
2.2. API level 31 (Android 12) és újabb . . . . .	2
<b>3. Heti munkák</b>	<b>3</b>
3.1. Első hét . . . . .	3
3.2. Második hét . . . . .	3
3.3. Harmadik hét . . . . .	4
3.4. Negyedik hét . . . . .	6
<b>4. Összegzés</b>	<b>7</b>

## **Kivonat**

A témalaboratóriumom során egy nyílt forráskódú Android alkalmazáshoz fejlesztettem egy töltőképernyő funkciót, hogy az az újabb, Android 12-t és újabbat futtató rendszereken is megfelelően működjön. Ehhez extenzíven megismerkedtem az Android dokumentációjával, példakódokat néztem, többféle megoldást teszteltem. A munkám során betekintést nyertem a közösségi szoftverfejlesztésbe, egy nagy, több, mint félmillió felhasználó eszközén futó program hatalmas kódbázisába.

## 1. fejezet

# Bevezetés, célkitűzések

Napjainkban egyre gyakrabban szembesülünk az egészséges életmód betartásának nehézségeivel. A témalaboratórium tárgy keretein belül olyan szoftverfejlesztési feladattal szerettem volna foglalkozni, ami kapcsolódik a választott témához (orvosi informatika) és gyakorlati jelentősége is van, a szoftver használatával megkönnyíthetjük, jobbá tehetjük a felhasználók életét.

Ebben az irányban elindulva, számos különböző projektet átnézve és a konzulensemmel egyeztetve döntöttem úgy, hogy a félév során az OpenFoodFacts Android alkalmazását fogom fejleszteni. Miután ez az elhatározás megszületett, igyekeztem olyan kihívást vállalni, ami megfelel a tárgy követelményeinek, tehát kellő nehézségű az implementációja, olyan részeket is tartalmaz, melyekkel új ismereteket szerezhetek, és mégis belefér a tárgy szűkös határidejeibe. Ezeket a szempontokat szem előtt tartva böngésztem a projekthez tartozó hibajelentéseket és egyéb felhasználói kéréseket, javaslatokat (ún. *feature requests*-eket). Így bukkantam rá egy olyan hibajegyre (*GitHub Issue*), amelyben az alkalmazás töltőképnyőjének (Splash screen) akkori implementációjának módosítását kérték, hogy az Android 31-es API szint feletti, azaz Android 12-t vagy annál újabbat futtató eszközökön is helyesen működjön.

Mivel korábban az egyetemen már volt lehetőségem betekintést nyerni a mobilos szoftverfejlesztésbe, és régebbi Android verziót futtató készülékekhez már készítettem hasonló töltőképnyőt, továbbá sok alkalmazásnál használnak ilyen megoldást, érdekelt, hogy miben újították meg a fejlesztés folyamatát az új API bevezetésével.

## 2. fejezet

# Androidos SplashScreen megoldások ismertetése

A továbbiakban ismertetett tevékenységek megértéséhez elengedhetetlen az Android SplashScreen megoldások rövid áttekintése.

### 2.1. API level 30 (Android 11) és korábbi

Az Android 12-ben bevezetett új API előtt nem volt rendszerszintű támogatás töltőképernyők fejlesztésére, így a fejlesztők két lehetőség közül választhattak:

- Egyedi téma bevezetése, amivel a nézet *windowBackground* propertyjét változtatták meg, majd állították vissza az alkalmazás betöltését követően az alapértelmezett értékre
- Egyedi *Activity* létrehozásával: ezzel egy dedikált osztályt és nézetet hozunk létre a töltőképernyő funkció megvalósítására, amely a betöltést vagy timeoutot követően elindítja az alkalmazás főképernyőjét

Az OpenFoodFacts alkalmazásban az utóbbit választották a fejlesztők, ez viszont azt eredményezte, hogy Android 11 fölötti eszközökön két különböző töltőképernyő jelent meg az alkalmazás indulásakor, emiatt vált szükségessé az alkalmazás felkészítése az újabb verzióval való helyes működésre.

### 2.2. API level 31 (Android 12) és újabb

A SplashScreen API bevezetésével maga az operációs rendszer biztosít egységes megoldást töltőképernyő készítésére. Ez olyan lehetőségekkel bővítette a programozók eszköztárát, ami korábban nem, vagy csak körülményesen volt megvalósítható (pl. animált ikonok beállítása a töltőképernyőre). Az új API arra is lehetőséget ad, hogy a régebbi szoftververziót futtató eszközökön a SplashScreen compat könyvtár segítségével biztosítsuk a visszafele kompatibilitást.

## 3. fejezet

# Heti munkák

Munkám jelentős részét az Android dokumentációjának tanulmányozása, illetve a különböző lehetőségek tesztelése tette ki. Fontos volt még megismerni a projekt fejlesztésére vonatkozó irányelveket (pl. kódformázási konvenciók, pull requestek pontos leírása).

### 3.1. Első hét

A projekt meghatározását követően igyekeztem megismerni annak felépítését, kideríteni, hogy pontosan melyik modulokkal kell majd dolgoznom a töltőkép-ernyő frissebb verziójának elkészítéséhez. Ehhez klónoztam a projekt GitHub repositoryját, és az Android Studio fejlesztőkörnyezet segítségével részletesen tanulmányoztam a projekt struktúráját, a releváns kódrészleteket, beszereztem a függőségeket, és lefordítottam a programot.

### 3.2. Második hét

Android 12-es verziót futtató emulátor telepítését követően az Android fejlesztői dokumentációjában fellelhető migrációs útmutatóból tájékozódtam a további feladataimról. Az itt leírtakat követve a töltőkép-ernyő helyesen jelent meg az API 31-et futtató emulátoron. Ilyen funkció esetében azonban nem feledkezhetünk meg a visszafelé kompatibilitás támogatásáról sem, ezért szükséges volt az új API-t nem ismerő eszközökön is tesztelni a megoldást. A régebbi API verziót futtató eszközök esetében azt a megoldást választottam, hogy a már meglévő töltőkép-ernyő jelenjen meg (azaz egy külön splash screen activity induljon el az alkalmazás indulásakor), míg az újakon a frissen implementált verzió. A funkció tesztelése során megállapítottam, hogy a megoldás alapvetően az elvárt eredményt adja, azonban – ahogy az a GitHub issue-ban is szerepelt – az éjszakai módban nem az elvárt módon működik: sötét helyett fehér háttérszínnel jelenik meg a képernyő. A hibajegyhez mellékelt képernyőképen szereplőhöz hasonló kinézetű töltőkép-ernyőt az emulátoron nem sikerült reprodukálni, ott az éjszakai mód bekapcsolását követően is fehér háttérszínnel indult el a töltőkép-ernyő, csak az alkalmazás további részei használták az éjszakai témát. Mivel Android 12-t futtató eszköz nem állt rendelkezésemre, ezért valódi hardveren nem tudtam tesztelni a megoldást.

A heti munkám során alkalmam nyílt megismerni egy, az Androidos szoftver-fejlesztésben elterjedt megoldást az alkalmazás különböző verzióinak egyszerű előállítására. Ebben a projektben ez különösképpen nagy jelentőséggel bír, mert nem csak arról van szó, hogy az alkalmazást több alkalmazásboltban is terjesztik (pl. Google Play, F-Droid), hanem ugyanarra a kódbázisra három különböző alkalmazást (OpenFoodFacts (OFF), OpenBeautyFacts (OBF), OpenPetFoodFacts (OPF)) is építenek. Ez azért is lehetséges, mert az alkalmazás lényegében csak egy felhasználói felület az OpenFoodFacts (és a többi verzió) adatbázisának elérésére. Így maga a kliens oldali alkalmazás az ikonok és egyéb felhasználói elemek kivételével megegyezik a három verzió között. A build variants

### 3.3. Harmadik hét

Ezen a héten a munkám az éjszakai módban látható töltőképnyő helyes megjelenítésére irányult. Sikertől kölcsönkértem egy Android 12-es verziót futtató eszközt, azonban ezen sem sikerült előállítanom a GitHub issue-ban szereplő módon a hibát, így folytattam a hiba okának felkutatását a kódban és az interneten egyaránt. Rövid utánajárást követően kiderült, hogy a *styles.xml* fájlból nem volt megadva *night* erőforrás-módosítóval (resource modifier) ellátott verzió. Ezt a hiányosságot pótoltam, létrehoztam egy éjszakai módban helyesen megjelenő stílus erőforrást. Ebben a háttérszín megváltoztatásán túl szükséges volt a SplashScreen API-által biztosított témából egy sajátot leszármaztatni, mely rendelkezik az ikon helyes megjelenítéséhez szükséges propertykkel. Az így született téma az alábbi:

```
<style name="SplashTheme"
    parent="Theme.SplashScreen.IconBackground">
    <item name="windowSplashScreenBackground">
        @color/grey_800
    </item>
    <item name="windowSplashScreenIconBackgroundColor">
        @color/grey_400
    </item>
    <item name="postSplashScreenTheme">
        @style/Theme.AppCompat.DayNight
    </item>
    <item name="windowSplashScreenAnimatedIcon">
        @mipmap/ic_launcher_round
    </item>
</style>
```

Ez azonban nem jelentett megoldást, mert a téma, amiből az API 30 feletti helyes működésért le kell származtatnunk a saját témánkat, nem támogatja az AppCompatActivity használatát, ami az OpenFoodFacts activity-implementációjának őosztálya. Így a SplashScreen API által biztosított téma kizárólagos használatával az alkalmazás a töltőképnyő megjelenítését követően crashelt, ez a 3.1 ábrán látható.

```

2022-10-30 19:43:00.107 8078-8078/? E/AndroidRuntime: FATAL EXCEPTION: main
Process: openfoodfacts.github.scrachx.openfood.debug, PID: 8078
java.lang.RuntimeException: Unable to start activity ComponentInfo{openfoodfacts.github.scrachx.openfood.debug/openfoodfacts.github.scrachx.openfood.features.splash.SplashActivity}: java.lang.IllegalStateException: You need to use a Theme.AppCompat theme (or descendant) with this activity.

```

3.1. ábra. *AppCompat* téma használata nélkül az alkalmazás összeomlik

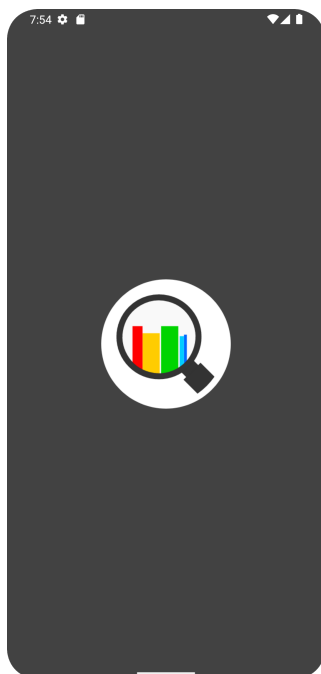
A probléma megoldására a StackOverflow-n és az Android fejlesztői oldalon elérhető leírás nyújtott iránymutatást. A `postSplashScreenTheme` property megfelelő beállítását követően az alkalmazás már megfelelően működött, a töltőképnyő után a főképernyő helyesen jelent meg. Így az éjszakai módban használt téma kódja az alábbi:

```

<style name="SplashTheme"
    parent="Theme.SplashScreen.IconBackground">
    <item name="windowSplashScreenBackground">
        @color/grey_800
    </item>
    <item name="windowSplashScreenIconBackgroundColor">
        @color/grey_400
    </item>
    <item name="postSplashScreenTheme">
        @style/Theme.AppCompat.DayNight
    </item>
</style>

```

A fent definiált téma az alábbi módon jelenik meg a gyakorlatban (3.2 ábra):



3.2. ábra. A töltőképnyő éjszakai módban, Android 12-n



### 3.4. Negyedik hét

A negyedik héten kommunikációt folytattam az alkalmazás fejlesztőivel a pull request lehetséges javításairól, illetve ennek merge-öléséről. A módosításaim végül elfogadásra kerültek, és bekerültek az alkalmazás kódbázisába.

4. fejezet

Összegzés